ARMY RESEARCH LABORATORY

Calculating Atmospheric Conditions (Temperature, Pressure, Air Density, and Speed of Sound) Using C++

by Robert J. Yager

ARL-TN-543 **June 2013**

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

Army Research Laboratory
Aberdeen Proving Ground, MD 21005-5066

ARL-TN-543 **June 2013**

Calculating Atmospheric Conditions (Temperature, Pressure, Air Density, and Speed of Sound) Using C++

Robert J. Yager
Weapons and Materials Research Directorate, ARL

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED (From - To)
June 2013	Final	January 2013–March 2013

4. TITLE AND SUBTITLE
Calculating Atmospheric Conditions (Temperature, Pressure, Air Density, and Speed of Sound) Using C++

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Robert J. Yager

5d. PROJECT NUMBER
AH80

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
U.S. Army Research Laboratory
ATTN: RDRL-WML-A
Aberdeen Proving Ground, MD 21005-5066

8. PERFORMING ORGANIZATION REPORT NUMBER
ARL-TN-543

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT
This report presents a set of functions, written in C++, that can be used to calculate atmospheric conditions (temperature, pressure, air density, and speed of sound), as well as gravitational-field strength. The functions are based on the atmospheric model presented in *U.S. Standard Atmosphere, 1976* (National Aeronautics and Space Administration. *U.S. Standard Atmosphere, 1976*; NASA-TM-X-74335; U.S. Government Printing Office: Washington, DC, October 1976).

15. SUBJECT TERMS
atmosphere, temperature, pressure, air density, speed of sound, gravity, U.S. Standard Atmosphere, C++

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
REPORT	b. ABSTRACT	c. THIS PAGE			Robert J. Yager
Unclassified	Unclassified	Unclassified	UU	16	19b. TELEPHONE NUMBER (*Include area code*) 410-278-6689

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

Contents

List of Tables

Acknowledgments

The author would like to thank Mr. Benjamin Flanders of the U.S. Army Research Laboratory's Weapons and Materials Research Directorate. Mr. Flanders provided technical and editorial recommendations that improved the quality of this report.

INTENTIONALLY LEFT BLANK.

1. Introduction

This report presents a set of functions, written in C++, that can be used to calculate atmospheric conditions (temperature, pressure, air density, and speed of sound), as well as gravitational-field strength. The functions are based on the atmospheric model presented in *U.S. Standard Atmosphere, 1976*.[1]

A summary sheet is provided at the end of this report. It presents the yAtmosphere namespace, which contains the five functions that are described in detail in this report. Also presented is an example that can be used to recreate portions of the "Main Tables" presented in *U.S. Standard Atmosphere, 1976*.

2. Summary of Constants

The following is a list of constants that are used by the code presented in this report. All are from *U.S. Standard Atmosphere, 1976*. Page references refer to the source document.

$R^* = 8314.32 \text{ N m/(kmol K)}$	universal gas constant	(page 3)
$r_0 = 6{,}356{,}766 \text{ m}$	effective radius of the earth	(page 4)
$\gamma = 1.400$	specific heat ratio	(page 4)
$T_0 = 288.15 \text{ K}$	standard temperature at sea level	(page 4)
$g_0' = 9.80665 \text{ m}^2/(\text{s}^2\text{m}')$	geopotential constant	(page 8)
$g_0 = 9.80665 \text{ m/s}^2$	gravitational-field strength at sea level	(page 8)
$\Gamma = 1 \text{ m}'/\text{m}$	unit-converting constant	(page 8)
$M_0 = 28.9644 \text{ kg/kmol}$	mean molecular weight of air	(page 9)
$P_0 = 101325.0 \text{ Pa}$	standard pressure at sea level	(page 12)

[1] National Aeronautics and Space Administration. *U.S. Standard Atmosphere, 1976*; NASA-TM-X-74335; U.S. Government Printing Office: Washington, DC, October 1976.

3. Summary of Equations

The following is a list of equations that are used by the code presented in this report. All are from *U.S. Standard Atmosphere, 1976*. Equation numbers refer to the source document.

$$g = g_0\left(\frac{r_0}{r_0 + z}\right)^2$$ gravitational-field strength (equation 17)

$$H = \frac{\Gamma r_0 z}{r_0 + z}$$ geopotential height (equation 18)

$$T_M = T_{M,b} + L_{M,b}(H - H_b)$$ molecular-scale temperature (equation 23)

$$P = \begin{cases} P_b\left[\dfrac{T_{M,b}}{T_{M,b} + L_{M,b}\left(H - H_b\right)}\right]^{\frac{g_0' M_0}{R^* L_{M,b}}} & \text{for } L_{M,b} \neq 0 \\ P_b e^{\frac{-g_0' M_0 (H - H_b)}{R^* T_{M,b}}} & \text{for } L_{M,b} = 0 \end{cases}$$ pressure (equation 33ab)

$$\rho = \frac{P M_0}{R^* T_M}$$ air density (equation 42)

$$c_{sound} = \left(\frac{\gamma R^* T_M}{M_0}\right)^{1/2}$$ speed of sound (equation 50)

4. Augmented Table 4

Along with equations 23 and 33ab, *U.S. Standard Atmosphere, 1976* specifies the use of tabulated values to calculate temperature and pressure. Values for H_b and $L_{M,b}$ are listed in the source document's table 4. Values for $T_{M,b}$ and P_b can be found by treating equations 23 and 33ab as recursive, with T_0 and P_0 from section 2 used as initial values. In practice, it is computationally efficient to precalculate values for $T_{M,b}$ and P_b. Table 1 lists the values for H_b and $L_{M,b}$ that were originally listed in the source document's table 4, as well as precalculated values for $T_{M,b}$ and P_b.

Table 1. Tabulated values for H_b, $L_{M,b}$, $T_{M,b}$, and P_b.

b	H_b (m')	$L_{M,b}$ (K/m')	$T_{M,b}$ (K)	P_b (Pa)
0	0	-0.0065	288.15	1.01325000000000E+05
1	11000	0	216.65	2.26320639734629E+04
2	20000	0.001	216.65	5.47488866967777E+03
3	32000	0.0028	228.65	8.68018684755228E+02
4	47000	0	270.65	1.10906305554966E+02
5	51000	-0.0028	270.65	6.69388731186873E+01
6	71000	-0.002	214.65	3.95642042804073E+00
7	84852	0	186.946	3.73383589976215E-01

5. Calculating Temperature and Pressure

The following method is valid for $-5{,}000\,\text{m} \le z < 86{,}000\,\text{m}$. Begin by calculating geopotential height (H) as a function of altitude (z) using equation 18. Next, find $b \mid H_b \le H < H_{b+1}$. If $H < 0\,\text{m}'$, use $b = 0$. If $H \ge 8{,}4852\,\text{m}'$, use $b = 7$.

To find molecular-scale temperature, substitute values for H_b, $L_{M,b}$, and $T_{M,b}$ into equation 23.

To find pressure, substitute values for H_b, $L_{M,b}$, $T_{M,b}$, and P_b into equation 33ab.

6. C++ Implementation

6.1 TABLE4 Array

The values for H_b, $L_{M,b}$, $T_{M,b}$, and P_b are stored in the TABLE4 array.

TABLE4 Code

```
const double TABLE4[8][4]={//<==============================TRANSITION POINTS
  00000  , -0.0065 , 288.150 , 1.01325000000000E+5 ,//          FOR PRESSURE &
  11000  ,  0.0000 , 216.650 , 2.26320639734629E+4 ,//          TEMPERATURE VS
  20000  ,  0.0010 , 216.650 , 5.47488866967777E+3 ,//            GEOPOTENTIAL
  32000  ,  0.0028 , 228.650 , 8.68018684755228E+2 ,//         ALTITUDE CURVES
  47000  ,  0.0000 , 270.650 , 1.10906305554966E+2 ,//               [table 4]
  51000  , -0.0028 , 270.650 , 6.69388731186873E+1 ,//          (3RD COLUMN IS
  71000  , -0.0020 , 214.650 , 3.95642042804073E+0 ,//            TEMPERATURE,
  84852  ,  0.0000 , 186.946 , 3.73383589976215E-1  //          4TH, PRESSURE)
};//~~~YAGENAUT@GMAIL.COM~~~~~~~~~~~~~~~~~~~~~~~~~~LAST~UPDATED~15MAR2013~~~~~~
```

6.2 Temperature() Function

The Temperature() function uses the method described in section 5 to calculate molecular-scale temperature as a function of altitude. For altitudes up to 80,000 m, molecular-scale temperature is equal to absolute temperature. At 80,000 m, molecular-scale temperatures begin to differ from absolute temperatures. At 86,000 m (the limit of this model), *U.S. Standard Atmosphere, 1976* states that the difference between molecular-scale temperature and absolute temperature is 0.0787 K.

Temperature() Code

```
inline double Temperature(//<==================================TEMPERATURE (K)
    double z){//<--------ALTITUDE (m) (T IS VALID FOR -5,000 m < z < 86,000 m)
  double H=z*6356766/(z+6356766);//.............................[equation 18]
  int b;/*<-*/for(b=0;b<7;++b)if(H<TABLE4[b+1][0])break;
  return TABLE4[b][2]+TABLE4[b][1]*(H-TABLE4[b][0]);//...........[equation 23]
}//~~~~YAGENAUT@GMAIL.COM~~~~~~~~~~~~~~~~~~~~~~~~~LAST~UPDATED~15MAR2013~~~~~~
```

Temperature() Parameters

z **z** is a height above sea level (in meters). Output of the Temperature() function is valid for –5000 m ≤ **z** ≤ 86,000 m. Output for **z** = –6356766 m (i.e., the center of the earth) is undefined.

Temperature() Return Value

The return value of the Temperature() function is the molecular-scale temperature, in Kelvin, at the given altitude (**z**). See the beginning of this section for a note on the minor difference between molecular-scale temperature and absolute temperature.

6.3 Pressure() Function

The Pressure() function uses the method described in section 5 to calculate atmospheric pressure as a function of altitude.

Pressure() Code

```
inline double Pressure(//<=======================================PRESSURE (Pa)
    double z){//<--------ALTITUDE (m) (P IS VALID FOR -5,000 m < z < 86,000 m)
  double H=z*6356766/(z+6356766);//.............................[equation 18]
  int b;/*<-*/for(b=0;b<7;++b)if(H<TABLE4[b+1][0])break;
  double C=-.0341631947363104;//...............C = -G0*M0/RSTAR [pages 8,9,3]
  double Hb=TABLE4[b][0],Lb=TABLE4[b][1],Tb=TABLE4[b][2],Pb=TABLE4[b][3];
  return Pb*(fabs(Lb)>1E-12?pow(1+Lb/Tb*(H-Hb),C/Lb):exp(C*(H-Hb)/Tb));
}//~~~~YAGENAUT@GMAIL.COM~~~~~~~~~~~~~~~~~~~~~~~~~LAST~UPDATED~15MAR2013~~~~~~
```

Pressure() Parameters

z **z** is a height above sea level (in meters). Output of the Pressure() function is valid for –5000 m ≤ **z** ≤ 86,000 m. Output for **z** = -6356766 m (i.e., the center of the earth) is undefined.

Pressure() Return Value

The return value of the Pressure() function is the pressure, in Pascals, at the given altitude (**z**).

6.4 Density() Function

The Density() function uses equation 42 to calculate air density as a function of temperature and pressure.

Density() Code

```
inline double Density(//<====================================DENSITY (kg/m^3)
    double T,//<-------------TEMPERATURE (K) (CALCULATE T USING Temperature())
    double P){//<-----------------PRESSURE (Pa) (CALCULATE P USING Pressure())
  return P*.0034836763559737 9/T;//.............................[equation 42]
}//~~~~YAGENAUT@GMAIL.COM~~~~~~~~~~~~~~~~~~~~~~~~~LAST~UPDATED~15MAR2013~~~~~~
```

Density() Parameters

T **T** is a local temperature (in Kelvin). Values for **T** can be determined using the Temperature() function. Output for **T** = 0 K is undefined.

P **P** is a local pressure (in Pascals). Values for **P** can be determined using the Pressure() function.

Density() Return Value

The return value of the Density() function is the local density of air, in kilograms/cubic meter.

6.5 SpeedofSound() Function

The SpeedofSound() function uses equation 50 to calculate speed of sound as a function of temperature.

SpeedofSound() Code

```
inline double SpeedofSound(//<============================SPEED OF SOUND (m/s)
    double T){//<------------TEMPERATURE (K) (CALCULATE T USING Temperature())
  return sqrt(401.87430086589*T);//.............................[equation 50]
}//~~~~YAGENAUT@GMAIL.COM~~~~~~~~~~~~~~~~~~~~~~~~~LAST~UPDATED~15MAR2013~~~~~~
```

SpeedofSound() Parameters

T **T** is a local temperature (in Kelvin). Values for **T** can be determined using the Temperature() function.

SpeedofSound() Return Value

The return value of the SpeedofSound() function is the local speed of sound, in meters per second.

6.6 Gravity() Function

The Gravity() function uses equation 17 to calculate the gravitational-field strength near the surface of the earth as a function of altitude.

Gravity() Code

```
inline double Gravity(//<==================ACCELERATION DUE TO GRAVITY (m/s^2)
    double z){//<--------ALTITUDE (m) (g IS VALID FOR -5,000 m < z < 86,000 m)
  return 9.80665*pow(1+z/6356766,-2);//.........................[equation 17]
}//~~~~YAGENAUT@GMAIL.COM~~~~~~~~~~~~~~~~~~~~~~~~~LAST~UPDATED~15MAR2013~~~~~~
```

Gravity() Parameters

z **z** is a height above sea level (in meters). The output of the Gravity() function is valid for –5000 m ≤ **z** ≤ 1,000,000 m. Output for **z** ≤ –6356766 m is undefined.

Gravity() Return Value

The return value of the Gravity() function is the gravitational-field strength, in meters per second squared.

---

7. Summary

A summary sheet is provided at the end of this report. It presents the yAtmosphere namespace, which contains the five functions described in detail in section 6. Also presented is an example that can be used to recreate portions of the "Main Tables" presented in *U.S. Standard Atmosphere, 1976.*[1]

Some values that are calculated using the example code differ from values presented in *U.S. Standard Atmosphere, 1976* in the least-significant figure. For example, at an altitude of 85,500 m, *U.S. Standard Atmosphere, 1976* states that the molecular-scale temperature will be 187.920 K, while the example code predicts that the molecular-scale temperature will be 187.919 K.

yAtmosphere Summary

y_atmosphere.h

```
#ifndef Y_ATMOSPHERE_H_
#define Y_ATMOSPHERE_H_
#include <cmath>//.....................................fabs(),exp(),pow(),sqrt()
namespace yAtmosphere{//<=============================================ATMOSPHERIC MODEL
  //@@@@@@@@@@@@@@@@ [BASED ON "U.S. STANDARD ATMOSPHERE, 1976"] @@@@@@@@@@@@@@@@
  const double TABLE4[8][4]={//<================================TRANSITION POINTS
    00000  , -0.0065 , 288.150 , 1.01325000000000E+5 ,//          FOR PRESSURE &
    11000  ,  0.0000 , 216.650 , 2.26320639734629E+4 ,//          TEMPERATURE VS
    20000  ,  0.0010 , 216.650 , 5.47488866967777E+3 ,//            GEOPOTENTIAL
    32000  ,  0.0028 , 228.650 , 8.68018684755228E+2 ,//         ALTITUDE CURVES
    47000  ,  0.0000 , 270.650 , 1.10906305554966E+2 ,//                [table 4]
    51000  , -0.0028 , 270.650 , 6.69388731186873E+1 ,//          (3RD COLUMN IS
    71000  , -0.0020 , 214.650 , 3.95642042804073E+0 ,//            TEMPERATURE,
    84852  ,  0.0000 , 186.946 , 3.73383589976215E-1  //          4TH, PRESSURE)
  };//~~~YAGENAUT@GMAIL.COM~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~LAST~UPDATED~15MAR2013~~~~~~
  inline double Temperature(//<=====================================TEMPERATURE (K)
      double z){//<--------ALTITUDE (m) (T IS VALID FOR -5,000 m < z < 86,000 m)
    double H=z*6356766/(z+6356766);//..............................[equation 18]
    int b;/*<-*/for(b=0;b<7;++b)if(H<TABLE4[b+1][0])break;
    return TABLE4[b][2]+TABLE4[b][1]*(H-TABLE4[b][0]);//............[equation 23]
  }//~~~~YAGENAUT@GMAIL.COM~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~LAST~UPDATED~15MAR2013~~~~~~
  inline double Pressure(//<========================================PRESSURE (Pa)
      double z){//<--------ALTITUDE (m) (P IS VALID FOR -5,000 m < z < 86,000 m)
    double H=z*6356766/(z+6356766);//..............................[equation 18]
    int b;/*<-*/for(b=0;b<7;++b)if(H<TABLE4[b+1][0])break;
    double C=-.0341631947363104;//................C = -G0*M0/RSTAR [pages 8,9,3]
    double Hb=TABLE4[b][0],Lb=TABLE4[b][1],Tb=TABLE4[b][2],Pb=TABLE4[b][3];
    return Pb*(fabs(Lb)>1E-12?pow(1+Lb/Tb*(H-Hb),C/Lb):exp(C*(H-Hb)/Tb));
  }//~~~~YAGENAUT@GMAIL.COM~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~LAST~UPDATED~15MAR2013~~~~~~
  inline double Density(//<=========================================DENSITY (kg/m^3)
      double T,//<-------------TEMPERATURE (K) (CALCULATE T USING Temperature())
      double P){//<-----------------PRESSURE (Pa) (CALCULATE P USING Pressure())
    return P*.0034836763559737 9/T;//..............................[equation 42]
  }//~~~~YAGENAUT@GMAIL.COM~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~LAST~UPDATED~15MAR2013~~~~~~
  inline double SpeedofSound(//<=============================SPEED OF SOUND (m/s)
      double T){//<------------TEMPERATURE (K) (CALCULATE T USING Temperature())
    return sqrt(401.87430086589*T);//..............................[equation 50]
  }//~~~~YAGENAUT@GMAIL.COM~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~LAST~UPDATED~15MAR2013~~~~~~
  inline double Gravity(//<====================ACCELERATION DUE TO GRAVITY (m/s^2)
      double z){//<--------ALTITUDE (m) (g IS VALID FOR -5,000 m < z < 86,000 m)
    return 9.80665*pow(1+z/6356766,-2);//..........................[equation 17]
  }//~~~~YAGENAUT@GMAIL.COM~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~LAST~UPDATED~15MAR2013~~~~~~
}//@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
#endif
```

Example

```
#include <cstdio>//.....................FILE,freopen(),stdout,printf(),fclose()
#include "y_atmosphere.h"//Temperature(),Pressure(),Density(),SpeedofSound(),...
int main(){
  using namespace yAtmosphere;
  FILE *f=freopen("atmosphere.txt","w",stdout);//................redirect output
  printf("#Z-Altitude | Temperature |    Pressure    |    Density    | Sound   "
    "|  Gravity\n");
  printf("#    (m)    |     (K)     |      (Pa)      |    (kg/m^3)   | (m/s)   "
    "|  (m/s^2)\n#");
  for(int i=0;i<79;++i)printf("-");
  for(int i=0;i<1811;++i){
    if(i%10==0)printf("\n");
    double z=-5000+i*50;/*->*/printf("%8.0f",z);//.................altitude (m)
    double T=Temperature(z);/*->*/printf("%15.3f",T);//..........temperature (K)
    double P=Pressure(z);/*->*/printf("%17.4E",P);//..............pressure (Pa)
    double rho=Density(T,P);/*->*/printf("%16.4E",rho);//.......density (kg/m^3)
    double Vs=SpeedofSound(T);/*->*/printf("%11.2f",Vs);//.....sound speed (m/s)
    double g=Gravity(z);/*->*/printf("%11.4f\n",g);}//...........gravity (m/s^2)
  fclose(f);
  return 0;
}//~~~~~~YAGENAUT@GMAIL.COM~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~LAST~UPDATED~15MAR2013~~~~~~
```

atmosphere.txt (created with the above “Example” code)

```
#Z-Altitude | Temperature |    Pressure    |    Density    | Sound   | Gravity
#    (m)    |     (K)     |      (Pa)      |    (kg/m^3)   | (m/s)   | (m/s^2)
#-------------------------------------------------------------------------------
   -5000         320.676      1.7776E+005     1.9311E+000     358.99     9.8221
   -4950         320.350      1.7682E+005     1.9228E+000     358.80     9.8219
   -4900         320.025      1.7587E+005     1.9145E+000     358.62     9.8218
   -4850         319.699      1.7493E+005     1.9062E+000     358.44     9.8216
   -4800         319.374      1.7400E+005     1.8980E+000     358.26     9.8215
   -4750         319.048      1.7307E+005     1.8898E+000     358.07     9.8213
   -4700         318.723      1.7214E+005     1.8816E+000     357.89     9.8212
   -4650         318.397      1.7122E+005     1.8734E+000     357.71     9.8210
   -4600         318.072      1.7030E+005     1.8653E+000     357.53     9.8209
   -4550         317.746      1.6939E+005     1.8572E+000     357.34     9.8207
                                                .
                                                .
                                                .
    -500         291.400      1.0748E+005     1.2849E+000     342.21     9.8082
    -450         291.075      1.0685E+005     1.2788E+000     342.02     9.8080
    -400         290.750      1.0622E+005     1.2727E+000     341.83     9.8079
    -350         290.425      1.0560E+005     1.2667E+000     341.63     9.8077
    -300         290.100      1.0498E+005     1.2607E+000     341.44     9.8076
    -250         289.775      1.0436E+005     1.2547E+000     341.25     9.8074
    -200         289.450      1.0375E+005     1.2487E+000     341.06     9.8073
    -150         289.125      1.0314E+005     1.2427E+000     340.87     9.8071
    -100         288.800      1.0253E+005     1.2368E+000     340.68     9.8070
     -50         288.475      1.0193E+005     1.2309E+000     340.49     9.8068
                                                .
                                                .
                                                .
       0         288.150      1.0133E+005     1.2250E+000     340.29     9.8066
      50         287.825      1.0073E+005     1.2191E+000     340.10     9.8065
     100         287.500      1.0013E+005     1.2133E+000     339.91     9.8063
     150         287.175      9.9536E+004     1.2075E+000     339.72     9.8062
     200         286.850      9.8945E+004     1.2017E+000     339.53     9.8060
     250         286.525      9.8358E+004     1.1959E+000     339.33     9.8059
     300         286.200      9.7773E+004     1.1901E+000     339.14     9.8057
     350         285.875      9.7191E+004     1.1844E+000     338.95     9.8056
     400         285.550      9.6611E+004     1.1786E+000     338.76     9.8054
     450         285.225      9.6035E+004     1.1729E+000     338.56     9.8053
                                                .
                                                .
                                                .
   85000         188.893      4.4568E-001     8.2195E-006     275.52     9.5496
   85050         188.796      4.4177E-001     8.1516E-006     275.45     9.5494
   85100         188.698      4.3790E-001     8.0843E-006     275.38     9.5493
   85150         188.601      4.3405E-001     8.0174E-006     275.31     9.5491
   85200         188.504      4.3024E-001     7.9511E-006     275.24     9.5490
   85250         188.406      4.2646E-001     7.8853E-006     275.16     9.5488
   85300         188.309      4.2271E-001     7.8201E-006     275.09     9.5487
   85350         188.212      4.1899E-001     7.7553E-006     275.02     9.5485
   85400         188.114      4.1531E-001     7.6910E-006     274.95     9.5484
   85450         188.017      4.1165E-001     7.6273E-006     274.88     9.5482

   85500         187.919      4.0802E-001     7.5640E-006     274.81     9.5481
```

NO. OF COPIES	ORGANIZATION
1 (PDF)	DEFENSE TECHNICAL INFORMATION CTR DTIC OCA
1 (PDF)	DIRECTOR US ARMY RESEARCH LAB IMAL HRA
1 (PDF)	DIRECTOR US ARMY RESEARCH LAB RDRL CIO LL
1 (PDF)	GOVT PRINTG OFC A MALHOTRA

ABERDEEN PROVING GROUND

NO. OF COPIES	ORGANIZATION
38 (33 PDF 5 HC)	DIR USARL RDRL EIS AM D ERICSON C DOUROS RDRL SLB W P GILLICH L HALL C KENNEDY N MOHOLKAR T MYERS K RAFAELS RDRL SSL T J BELL C EWING E LYNCH K LEWIS Y SOHN G WIGGS RDRL WM P BAKER RDRL WML M ZOLTOSKI RDRL WML A M ARTHUR B BREECH P BUTLER B FLANDERS W OBERLE C PATTERSON R PEARSON L STROHM A THOMPSON R YAGER (5 HC, 1 PDF) RDRL WML B N TRIVEDI
	RDRL WML C S AUBERT RDRL WML D R BEYER RDRL WML E P WEINACHT RDRL WML F D LYON RDRL WML G J SOUTH RDRL WML H J NEWILL